Lösningsförslag på tentamen från 121220

Uppgift 1 (7 + 2=9p)

a. Skriv en metod som skriver ut en sträng framlänges eller baklänges beroende på värdet av den andra parametern (den första är själva strängen). För att få strängen utskriven framlänges anropar man metoden med:
skrivStraeng("Hej och hå", "Fram");
och för att få strängen utskriven baklänges anropar man metoden med:
skrivStraeng("Hej och hå", "Bak");
b. Skriv ett testprogram som testar båda sätt att skriva ut strängen.

Vi börjar med a.
"*Skriv en metod som skriver ut en sträng*", en metod som skriver ut något och inte returnerar något! Kommer ni ihåg? void betyder ju "Bara gör något, returnera inget". Alltså vet vi att metoden ska deklareras med returtypen void. Metoden ska ha namnet skrivStraeng och ha två parametrar av typen String-referens. Vi kan alltså redan nu börja med en skiss av metoden:

```
void skrivStraeng(String str, String riktning){

}
```

Andra parametern kan vara antingen "Fram" eller "Bak" och metoden ska göra olika saker beroende på detta. Vi behöver alltså undersöka värdet på andra parametern (som jag kallade "riktning"):

```
if ( riktning.equals("Fram") ){
    // Skriv ut strängen framlänges, dvs skriv ut den som den är:
    System.out.println(str);
} else if( riktning.equals("Bak") ){
    // Skapa en ny sträng med str baklänges och skriv ut den
}
```

Nu har vi en mer komplett metod (som även om vi inte kommer längre torde generera några tröstpoäng på tentan ;-)):

```
void skrivStraeng(String str, String riktning){
    if ( riktning.equals("Fram") ){
        // Skriv ut strängen framlänges, dvs skriv ut den som den är:
        System.out.println(str);
    } else if( riktning.equals("Bak") ){
        // Skriv ut str baklänges
    }
}
```

Det knepiga här verkar vara att komma på ett sätt att skriva ut en sträng baklänges då, alltså om andra parametern riktning.equals("Bak").

Bilaga till tentan var dokumentationen av vissa metoder till String från Javas standard-API. Där finner vi några användbara metoder:
`char charAt(int index)` som returnerar en viss char på en viss position.
`int length()` som returnerar längden på strängen.
Vi utgår från att första index på en Strings tecken har index 0 (eftersom det förmodligen handlar om en array av char). Då måste sista index vara (`length() -1`).
Exempel: “ABCDE”

```
“ABCDE”.charAt(0) == ‘A’
“ABCDE”.charAt(1) == ‘B’
“ABCDE”.charAt(2) == ‘C’
“ABCDE”.charAt(3) == ‘D’
“ABCDE”.charAt(4) == ‘E’
“ABCDE”.length()  ==  5
```

Om vi köper detta resonemang, så kan vi skissa på en strategi för att skriva ut strängen baklänges!
Vi börjar med att skriva ut str.charAt(str.length() - 1) och så arbetar vi oss baklänges ned till str.charAt(0) ett index i taget. Hmm, detta liknar ju en for-loop!

```
int start = str.length() - 1;
for (int i = start; i >= 0; i--){
    System.out.print( str.charAt(i) ); // Använd print, ingen ny rad!
}
// Skriv en tom nyrad
System.out.println();
```

Nu kan vi pilla in vår strategi på rätt plats i else if-blocket:

```
void skrivStraeng(String str, String riktning){
    if ( riktning.equals("Fram") ){
        // Skriv ut strängen framlänges, dvs skriv ut den som den är:
        System.out.println(str);
    } else if( riktning.equals("Bak") ){
        // Skriv ut str baklänges
        int start = str.length() - 1; // Sista tecknets index
        for (int i = start; i >= 0; i--){
            System.out.print( str.charAt(i) ); // Använd print, ingen ny rad!
        }// For-loop
        // Skriv en tom nyrad
        System.out.println();
    }// else-if
}// metoden
```

Deluppgift b. är betydligt enklare: *Skriv ett testprogram som testar båda sätt att skriva ut strängen.*

Program i Java skriver man ju med en main-metod. Alla metoder skrivs ju i Java i en klass. Så vi börjar med att definiera klassen med metoden main och så vår nya metod skrivStraeng:

```
public class Uppgift1{
    public static void main(String[] args){
        // Testanrop av skrivStraeng här
    }

    void skrivStraeng(String str, String riktning){
        if ( riktning.equals("Fram") ){
            // Skriv ut strängen framlänges, dvs skriv ut den som den är:
            System.out.println(str);
        } else if( riktning.equals("Bak") ){
            // Skriv ut str baklänges
            int start = str.length() - 1;
            for (int i = start; i >= 0; i--){
                System.out.print( str.charAt(i) ); // Använd print!
            }// For-loop
            // Skriv en tom nyrad
            System.out.println();
        }// else-if
    }// metoden

}// Class
```

Testanrop av metoden kan vi sno från uppgiften i sig!
skrivStraeng("Hej och hå", "Fram");
skrivStraeng("Hej och hå", "Fram");

Så allt vi behöver göra är att slänga in de raderna i main!

Men vänta lite nu, så som skrivStraeng är definierad i klassen är det en instansmetod. Vill vi kunna anropa den från main, utan att skapa en instans av klassen, så måste vi göra om metoden till en klassmetod med static! Eller så får vi skapa en instans av klassen och anropa den via instansens referensvariabel.
Så här ser det kompletta programmet ut med metoden som klassmetod (static):

```
public class Uppgift1{
    public static void main(String[] args){
        // Testanrop av skrivStraeng här
        skrivStraeng("Hej och hå", "Fram");
        skrivStraeng("Hej och hå", "Bak");
    }

    static void skrivStraeng(String str, String riktning){
        if ( riktning.equals("Fram") ){
            // Skriv ut strängen framlänges, dvs skriv ut den som den är:
            System.out.println(str);
        } else if( riktning.equals("Bak") ){
            // Skriv ut str baklänges
            int start = str.length() - 1;
            for (int i = start; i >= 0; i--){
                System.out.print( str.charAt(i) ); // Använd print!
            }// For-loop
            // Skriv en tom nyrad
            System.out.println();
        }// else-if
    }// metoden

}// Class
```

Så här ser programmet ut om ni föredrar att ha metoden som en instansmetod och skapa en instans av klassen i main:

```
public class Uppgift1Instansvariant{
    public static void main(String[] args){
        // Testanrop av skrivStraeng här
        Uppgift1Instansvariant u = new Uppgift1Instansvariant();
        u.skrivStraeng("Hej och hå", "Fram");
        u.skrivStraeng("Hej och hå", "Bak");
    }

    void skrivStraeng(String str, String riktning){
        if ( riktning.equals("Fram") ){
            // Skriv ut strängen framlänges, dvs skriv ut den som den är:
            System.out.println(str);
        } else if( riktning.equals("Bak") ){
            // Skriv ut str baklänges
            int start = str.length() - 1;
            for (int i = start; i >= 0; i--){
                System.out.print( str.charAt(i) ); // Använd print!
            }// For-loop
            // Skriv en tom nyrad
            System.out.println();
        }// else-if
    }// metoden

}// Class
```

Överkurs

Vad händer om str eller riktning är null? Då kraschar programmet.

Om riktning är null kraschar programmet redan vid första raden:

```
if ( riktning.equals("Fram") )
```

Om ni anropar en instansmetod, t ex equals() på null så får ni ett NullPointerException och programmet kraschar.

Om str är null så kraschar programmet när ni anropar instansmetoden length(). Det är inte bra.

Så vi bör hantera att en eller båda parametrarna kan vara null. Det finns mängder med sätt att hantera det, jag ger ett förslag här som går ut på att om någon av parametrarna är null så returnerar metoden helt enkelt utan att göra något. Även om metoden är deklarerad void så går det att ha en return-sats som bara skriver `return;` Det är fortfarande ingenting som verkligen returneras, en tom return; i en void-metod bara avslutar metoden. Man kan tänka att exekveringen hoppar ut från metoden om en sådan sats nås.

Allt vi behöver lägga till i vår metod är nu att vi börjar med att testa om någon av parametrarna är null och i så fall returnera utan att göra ett enda dugg.

```
if ( str == null || riktning == null ){
    return;
}
```

Lägg denna sats först i metoden om ni vill testa att den ska tåla att bli anropad med en av parametrarna som null utan att den ska krascha programmet. Nästa sida innehåller ett sådant exempel!

```
public class Uppgift1NullSafe{
    public static void main(String[] args){
        skrivStraeng(null, "Fram");
        skrivStraeng("Hej och hå", null);
        skrivStraeng("Hej och hå", "Fram");
        skrivStraeng("Hej och hå", "Bak");
    }

    static void skrivStraeng(String str, String riktning){
        if ( str == null || riktning == null ){
            return;
        }
        if ( riktning.equals("Fram") ){
            // Skriv ut strängen framlänges:
            System.out.println(str);
        } else if( riktning.equals("Bak") ){
            // Skriv ut str baklänges
            int start = str.length() - 1;
            for (int i = start; i >= 0; i--){
                System.out.print( str.charAt(i) ); // Använd print!
            }// For-loop
            // Skriv en tom nyrad
            System.out.println();
        }// else-if
    }// metoden

}// Class
```